

PROJECT

You are a DevOps engineer at XYZ Ltd. Your company is working on a Java application and wants to automate WAR file artifact deployment so that they don't have to perform WAR deployment on Tomcat/Jetty web containers. Automate Ansible integration with Jenkins CI server so that we can run and execute playbooks to deploy custom WAR files to a web container and then perform restart for the web container.

1. Configure Jenkins server as Ansible provisioning machine
2. Install Ansible plugins in Jenkins CI server
3. Prepare Ansible playbook to run Maven build on Jenkins CI server
4. Prepare Ansible playbook to execute deployment steps on the remote web container with restart of the web container post deployment

Solution:-

1. On the ACM (ansible controller machine), Connect the master node to the worker nodes by copying ssh-key on worker nodes. Add Public ip of worker nodes in the ansible host file and run the ping command to check the connectivity.

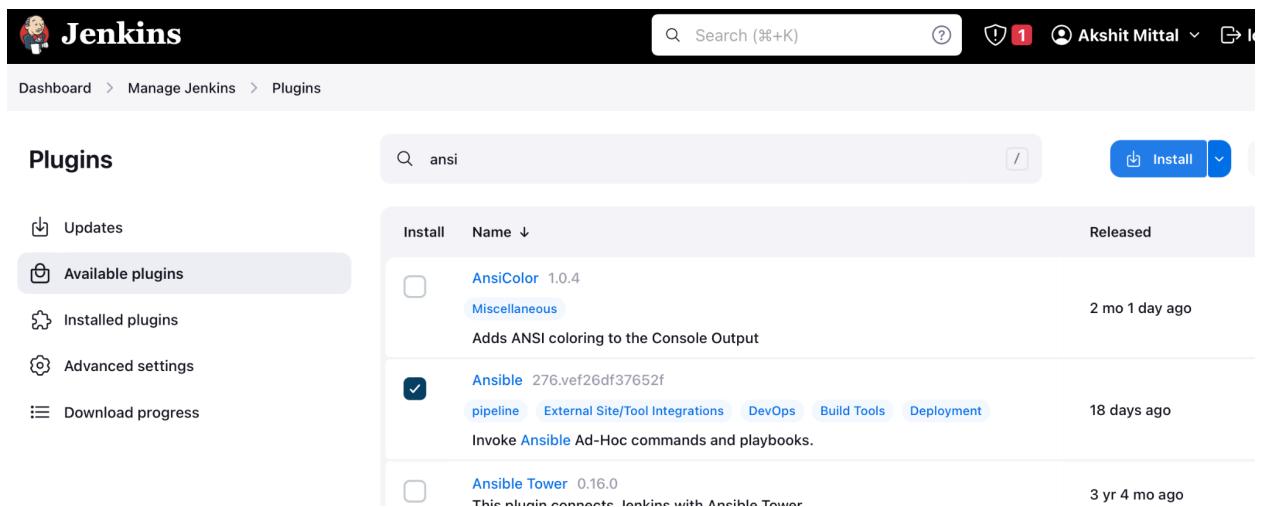
```
[akshit@AKSHITs-MacBook-Pro ~ % ssh-copy-id akshit@54.156.59.164
The authenticity of host '54.156.59.164 (54.156.59.164)' can't be established.
ED25519 key fingerprint is SHA256:PeTgmQrSURYZ6U+lg++YcWSSvpkaS77qK1N4eINTIj8.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out
any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
[akshit@54.156.59.164's password:

Number of key(s) added:          1

Now try logging into the machine, with:    "ssh 'akshit@54.156.59.164'"
and check to make sure that only the key(s) you wanted were added.]
```

```
[akshit@AKSHITs-MacBook-Pro ansible % vim hosts
[akshit@AKSHITs-MacBook-Pro ansible % ansible -i hosts webserver -m ping
54.156.59.164 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
[akshit@AKSHITs-MacBook-Pro ansible % cat hosts
[webserver]
54.156.59.164
```

- On the Controller Machine, Install Java, jenkins. Complete the setup. Start the jenkins service, and go on the plugins section to install Ansible and then → Tools section to add Ansible in tools with ansible-playbook path in the path section of ansible

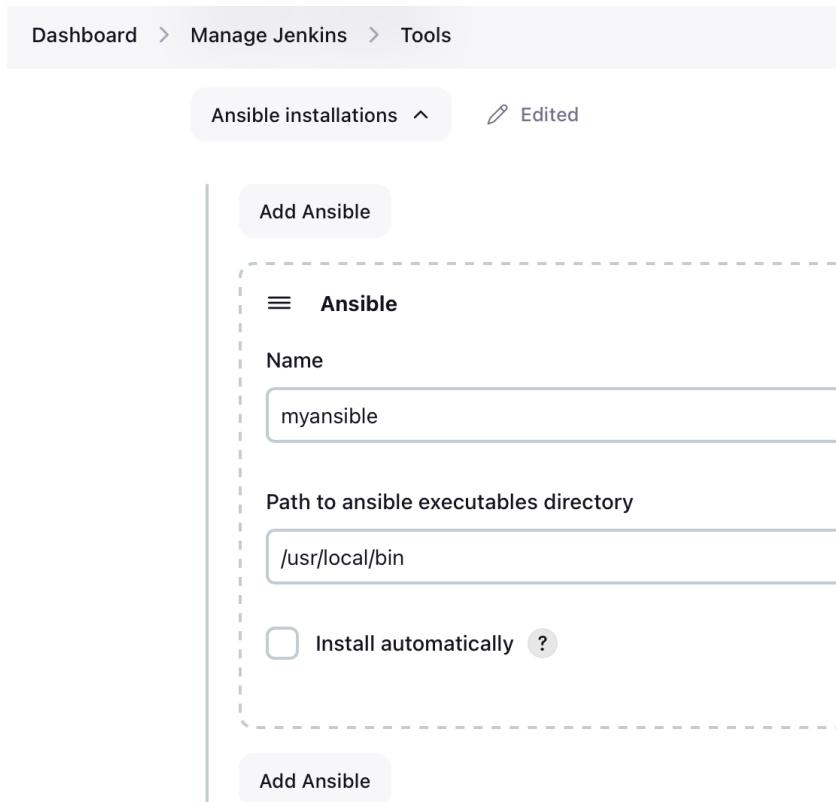


The screenshot shows the Jenkins Plugins page. The search bar at the top contains the text 'ansi'. Below the search bar, there is a table listing three plugins:

Install	Name	Released
<input type="checkbox"/>	AnsiColor 1.0.4 Miscellaneous	2 mo 1 day ago
<input checked="" type="checkbox"/>	Ansible 276.vef26df37652f pipeline External Site/Tool Integrations DevOps Build Tools Deployment	18 days ago
<input type="checkbox"/>	Ansible Tower 0.16.0 This plugin connects Jenkins with Ansible Tower	3 yr 4 mo ago

On the left side of the page, there is a sidebar with the following options:

- Updates
- Available plugins** (selected)
- Installed plugins
- Advanced settings
- Download progress



3. Now our task is to write 2 playbooks- 1 for installation of packages and Another playbook for deployment of code and building the Artifact and deploy on GitHub to be called via jenkins at the time of pipeline run.
But here,
First we will try to do it on our Controller machine, if the playbooks are running fine, then we will deploy the playbooks and Inventory file on Github and call via jenkins.

→ So, in the Ansible directory, we will be having 3 files

```
[akshit@AKSHITs-MacBook-Pro ansible % ls  
deploymentPlaybook.yml           installationPlaybook.yml  
hosts
```

→ In the Installation Playbook

```
- name: Installation packages playbook
hosts: webserver
become: true
become_user: root
tasks:
- name: update the apt-get repo
  command: sudo apt-get update
- name: Install maven on worker nodes
  package:
    name=maven
    state=present
- name: Install docker using module
  package:
    name=docker.io
    state=present
- name: Start docker service
  service:
    name=docker
    state=started
```

```
~  
~
```

```
[akshit@AKSHITs-MacBook-Pro ansible % ansible-playbook -i hosts installationPlaybook.yml

PLAY [Installation packages playbook] ****
TASK [Gathering Facts] ****
ok: [54.156.59.164]

TASK [update the apt-get repo] ****
changed: [54.156.59.164]

TASK [Install maven on worker nodes] ****
ok: [54.156.59.164]

TASK [Install docker using module] ****
changed: [54.156.59.164]

TASK [Start docker service] ****
ok: [54.156.59.164]

PLAY RECAP ****
54.156.59.164 : ok=5    changed=2    unreachable=0    failed=0    skipped=0    re
```

The playbook is executing successfully

→ In the deployment playbook

```
- name: Playbook to clone, build, deploy the java code
  hosts: webserver
  become: true
  become_user: root
  tasks:
    - name: clone the repo with java code on every worker node
      git:
        repo=https://github.com/Sonal0409/DevOpsCodeDemo.git
        dest=/tmp/devopscode-repo
    - name: Maven to build the code
      command: chdir=/tmp/devopscode-repo mvn package
    - name: From target folder copy the addressbook.war in current directory
      copy:
        src=/tmp/devopscode-repo/target/addressbook.war
        dest=/tmp/devopscode-repo
        remote_src=yes
    - name: Build the dockerfile
      command: chdir=/tmp/devopscode-repo docker build -t mycodeimage .
    - name: Run the image to deploy the application on tomcat container
      command: docker run -d -P mycodeimage
```

- We have cloned the git repo in the folder
- Using the maven tool, we build the code via package module
- We saw that the build executable file with extension “.war” was not present in the same folder but in the target folder names addressbook.war, so we copied the file into the main folder
- We run the docker command to build the image of our file
- We run the image to build the container running on the server as seen in the screenshots below.

```
lakshit@AKSHITs-MacBook-Pro:~/git/deployment% vim deploymentPlaybook.yml
lakshit@AKSHITs-MacBook-Pro ansible % ansible-playbook -i hosts deploymentPlaybook.yml

PLAY [Playbook to clone, build, deploy the java code] ****
TASK [Gathering Facts] ****
ok: [54.156.59.164]

TASK [clone the repo with java code on every worker node] ****
changed: [54.156.59.164]

TASK [Maven to build the code] ****
changed: [54.156.59.164]

TASK [From target folder copy the addressbook.war in current directory] ****
changed: [54.156.59.164]

TASK [Build the dockerfile] ****
changed: [54.156.59.164]

TASK [Run the image to deploy the application on tomcat container] ****
changed: [54.156.59.164]

PLAY RECAP ****
54.156.59.164 : ok=6    changed=5    unreachable=0    failed=0    skipped=0    re
scued=0    ignored=0
```

The .war deployable file has been copied into the main folder

```
lakshit@AKSHITs-MacBook-Pro ansible % ansible -i hosts webserver -m command -a "ls /tmp/devopscode-repo"
54.156.59.164 | CHANGED | rc=0 >>
CICD-pipeline-complete
Jenkinsfile
Newfiletrigger
README.md
addressbook.war
addressbook_screenshot.png
build.properties
build.xml
demofile
deployment.yml
dockerfile
jenkinsfile2
kubedeploy.yml
newcommit
newdemofile
pom.xml
src
target
```

The image has been built successfully and the container is running successfully.

```
lakshit@AKSHITs-MacBook-Pro ansible % ansible -i hosts webserver -m command -a "sudo docker images"
54.156.59.164 | CHANGED | rc=0 >>
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
mycodeimage     latest    19313eeec778  13 minutes ago  478MB
tomcat          9        3d0efbe5f648  7 days ago   462MB
```

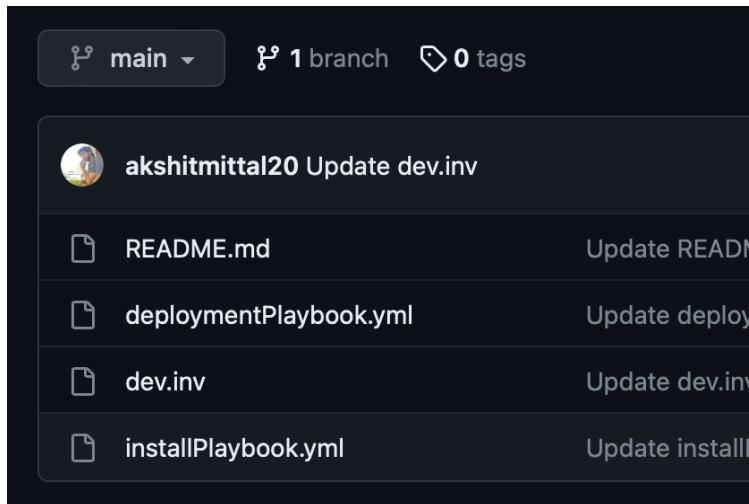
```
[akshit@AKSHITs-MacBook-Pro ansible % ansible -i hosts webserver -m command -a "sudo docker ps -a"
54.156.59.164 | CHANGED | rc=0 >>
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS
 NAMES
3b8fa98352cb      mycodeimage        "catalina.sh run"   8 minutes ago     Up 8 minutes      0.0.0.0:32768->8080/tcp, :::
32768->8080/tcp   keen_poincare

[akshit@AKSHITs-MacBook-Pro ansible % ansible -i hosts webserver -m command -a "sudo docker images"]
```

Now, we will do this via automation using Jenkins.

4. We will copy all 3 files on the git repo , which will be then called in the jenkins pipeline.

```
[akshit@AKSHITs-MacBook-Pro ansible % ls
deploymentPlaybook.yml      hosts
[akshit@AKSHITs-MacBook-Pro ansible % cat hosts
[webserver]
54.156.59.164
```



→ We can take the help of pipeline syntax feature to generate the pipeline

The screenshot shows the 'Pipeline Syntax' page in a CI/CD tool. At the top, there's a navigation bar: Dashboard > JavaCodeDeployCICD > Pipeline Syntax. Below the navigation, there are sections for 'Credentials' (set to 'none') and two checkboxes: 'Include in polling?' (checked) and 'Include in changelog?' (checked). A large blue button labeled 'Generate Pipeline Script' is centered. Below it, a code block displays the generated pipeline script:

```
git branch: 'main', url: 'https://github.com/akshitmittal20/CICD_Java'
```

→ Use ansiblePlaybook; Invoke module for generating ansible playbook syntax

The screenshot shows the 'Pipeline Syntax' page with the 'Steps' section expanded. On the left, there are four links: 'Steps Reference', 'Global Variables Reference', 'Online Documentation', and 'Examples Reference'. On the right, under the 'Steps' heading, a list of steps is shown, each with a question mark icon. One step, 'ansiblePlaybook: Invoke an ansible playbook', is highlighted with a blue border.

Steps Reference	Steps
Global Variables Reference	Sample Step
Online Documentation	ansiblePlaybook: Invoke an ansible playbook
Examples Reference	ansiblePlaybook

→ Give credentials of controller machine username and ssh-key in the ssh credentials provider tab. Select Kind as “SSH username with private key” → Give your username in which ssh key is generated and ssh private key in the passphrase tab.

You can find ssh private key in controller machine using command `#cat ~/.ssh/id_rsa`

Jenkins Credentials Provider: Jenkins

Add Credentials

Domain
Global credentials (unrestricted)

Kind
SSH Username with private key

Scope ?
Global (Jenkins, nodes, items, all child items, etc)

ID ?
/akshit

Username
akshit

Treat username as secret ?

Private Key
 Enter directly

Passphrase
.....

Add **Cancel**

```
At most one of -i, -u, -p or -e should be used
|akshit@AKSHITs-MacBook-Pro ~ % cat ~/.ssh/id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3B1bnNzaC1rZXktdjEAAAAABG5vbmlUAAAAEb9uZQAAAAAAAAABAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAtnVDQiPQfR++ibDKpkJRD4mStGSxSE2Sp5gLUHzyz0BLdkHSp0z/
U560eA0kvRJKqkDItIgI9UiH34/Al5ITI1iL2a8Qte2YeE9WjGgtbh9PmyGNF/K11jYEd1
qKryRcbE0pUFFFo3Ph1M/FeISrIxVe9riw2aKNScvJ/V1i+QWQxdierFLd2bTVGwdNRVFA
j59bF8Ds1+4kLxndv0BLz1R3JALp5d0s7YAWRC5lhmWX6I0MIPN7aLKS+/30vbp/y0VPiR
NCuYEpOZojHE9+l1Cou+iIbFExrnRifgIQivVjdGFNk+gss+4hn90Zm4KwMhwk8wasyb2+
I40+Y4B70bpC/YroRfSVc3RWi1kxLWv0pV3t8nsONESkTtxQMs+0kYclX9SR1ri1YGmy8L
```

→ give name of repo file, for ip address, and installation playbook

Sample Step

ansiblePlaybook: Invoke an ansible playbook

ansiblePlaybook ?

Ansible tool

myansible

Playbook file path in workspace

deploymentPlaybook.yml

Inventory file path in workspace

dev.inv

SSH connection credentials

akshit (akshit)

+ Add ▾

→ Disable the host key check. And generate the script

Disable the host SSH key check

Colorized output

Extra parameters

Generate Pipeline Script

```
ansiblePlaybook credentialsId: 'Akshit-id', disableHostKeyChecking: true, installation: 'myansible', inventory: 'dev.inv', playbook: 'installPlaybook.yml', vaultTmpPath: ''
```

→ Similarly do the same steps for deployment playbook

→ Finally your pipeline will look like this

```
pipeline {
    agent any

    stages {
        stage('Clone a repo') {
            steps {
                git branch: 'main', url: 'https://github.com/akshitmittal20/CICD_JavaApplicationDeployment_Ansible_Docker_Jenkins.git'
            }
        }

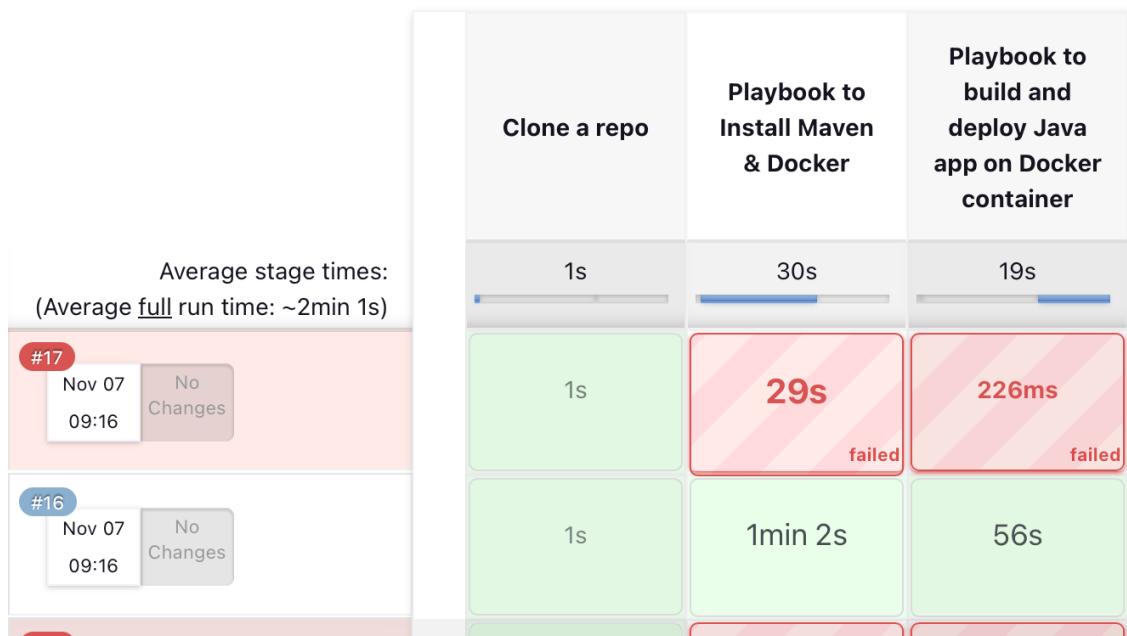
        stage('Playbook to Install Maven & Docker') {
            steps {
                ansiblePlaybook credentialsId: 'akshit',
                disableHostKeyChecking: true,
                installation: 'myansible',
                inventory: 'dev.inv',
                playbook: 'installPlaybook.yml'
            }
        }

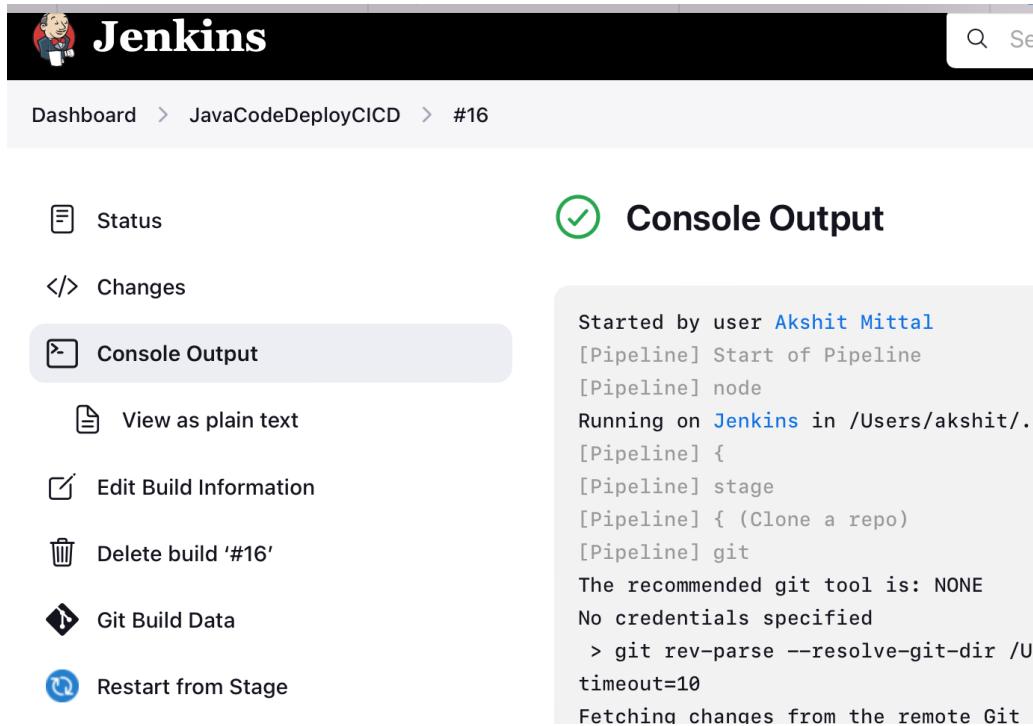
        stage('Playbook to build and deploy Java app on Docker container') {
            steps {
                ansiblePlaybook credentialsId: 'akshit',
                disableHostKeyChecking: true,
                installation: 'myansible',
                inventory: 'dev.inv',
                playbook: 'deploymentPlaybook.yml'
            }
        }
    }
}
```

→ Generate the output

Click on Build now and generate the output

Stage View





The screenshot shows the Jenkins interface for a JavaCodeDeployCICD project. The top navigation bar includes the Jenkins logo, a search bar, and a user icon. Below the header, the breadcrumb navigation shows 'Dashboard > JavaCodeDeployCICD > #16'. On the left, a sidebar lists options: Status, Changes, Console Output (which is selected and highlighted in grey), View as plain text, Edit Build Information, Delete build '#16', Git Build Data, and Restart from Stage. The main content area is titled 'Console Output' with a green checkmark icon. It displays the log output for build #16, which starts with 'Started by user Akshit Mittal' and continues with the pipeline stages: Start of Pipeline, node, Running on Jenkins in /Users/akshit/.jenkins/workspace/JavaCodeDeployCICD, Pipeline, stage, Clone a repo, git, and finally fetching changes from the remote Git.

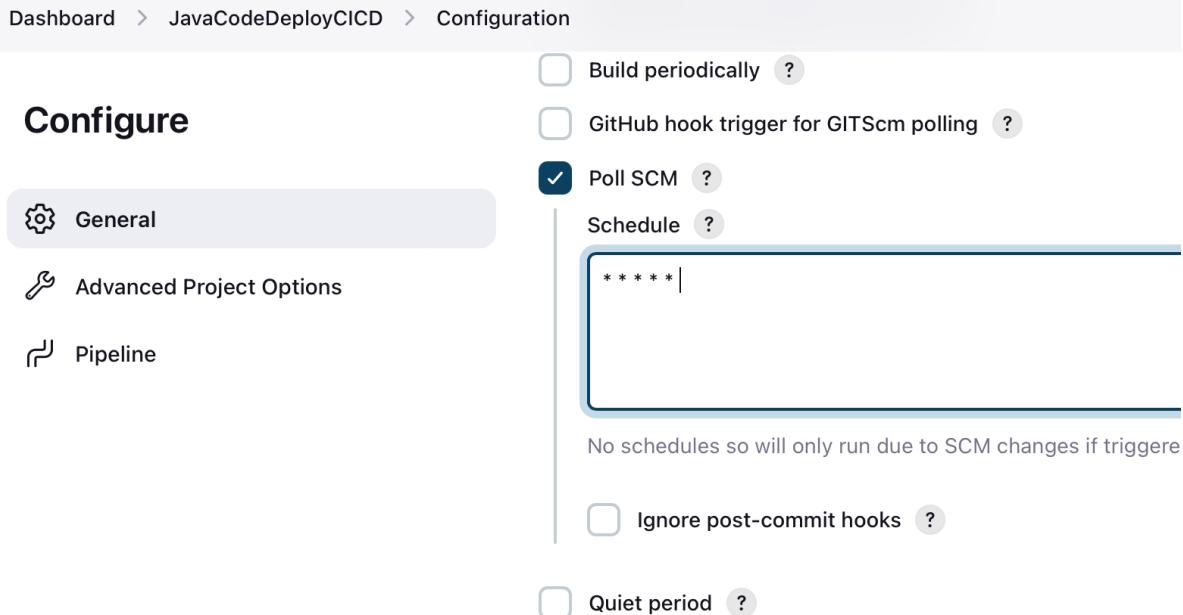
```

Started by user Akshit Mittal
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /Users/akshit/.jenkins/workspace/JavaCodeDeployCICD
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Clone a repo)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /Users/akshit/.jenkins/workspace/JavaCodeDeployCICD
timeout=10
Fetching changes from the remote Git ...

```

Pipeline is Successfully running.

For continuous deployment automation setup , we can set up the triggers in Jenkins, and it will check after sometime if any new commit is there, and pipeline will get triggered, via function Poll SCM.



The screenshot shows the Jenkins Configuration page for the JavaCodeDeployCICD project. The top navigation bar shows 'Dashboard > JavaCodeDeployCICD > Configuration'. The main title is 'Configure'. On the left, a sidebar lists General (selected and highlighted in grey), Advanced Project Options, and Pipeline. In the configuration area, under the 'General' section, there are several trigger options: 'Build periodically' (unchecked), 'GitHub hook trigger for GITScm polling' (unchecked), 'Poll SCM' (checked with a blue checkmark), and 'Schedule' (unchecked). The 'Poll SCM' section contains a text input field with the value '* * * * *'. A note below the schedule says 'No schedules so will only run due to SCM changes if triggered'. Other options shown include 'Ignore post-commit hooks' (unchecked) and 'Quiet period' (unchecked).

=====